Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Platforma pro srovnávací testy virtualizačních nástrojů

Benchmarking of Virtualization Tools

2014 **Jiří Pijáček**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Jiří Pijáček**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Platforma pro srovnávací testy virtualizačních nástrojů
Benchmarking of Virtualization Tools

Zásady pro vypracování:

Virtualizace je jednou z oblastí, která v současnosti zažívá výrazný rozmach v datových centrech i u koncových uživatelů. Cílem této práce bude v konzistentní formě dávkově otestovat existující virtualizační řešení, vyhodnotit výsledky testů a doporučit konkrétní řešení pro typické scénáře nasazení.

1. Nastudujte virtualizační techniky a požadavky pro běh virtualizačních nástrojů. Definujte základní pojmy (hypervisor, paravirtualizace, částečná virtualizace, aplikační virtualizace).

2. Srovnejte provoz aplikací ve virtualizovaném prostředí s provozem bez virtualizace, uveďte výhody a nevýhody obou řešení. Porovnejte dostupná řešení existujících výrobců (Citrix XenDesktop, Oracle VirtualBox, VMware vSphere/Workstation/ESXi, Microsoft Hyper-V, KVM) a vyberte alespoň dvě až tři technologicky dostatečně odlišná řešení pro další testy a nasaďte je.

3. Vytvořte platformu pro automatizované testování (benchmarking) jednotlivých komponent hostovaných systémů a sběr výsledků testů (prozkoumejte, jaké nástroje by bylo možné pro její tvorbu využít, a zda některý z nich Vašim požadavkům vyhovuje). Vytvořená platforma bude poskytovat zejména výkonnostní testy virtualizovaných CPU, paměti, diskových jednotek a síťové komunikace.

4. Proveďte testy na vybraných platformách v různých konfiguračních režimech (s a bez podpory VT-X/AMD-V), pro jednotlivé podporované typy virtuálních disků a režimu síťové komunikace (přímé připojení, most, NAT ve virt. nástroji), a bude-li to možné i grafický benchmark pro 2D a 3D grafiku včetně varianty s akcelerací grafických operací virtualizačním nástrojem, je-li podporována.

5. Zhodnoťte výsledky testů, zhodnoťte vhodnost platforem pro konkrétní použití (server, kancelářské PC, herní PC, apod.) a uveďte, jak Vaši práci dále rozšířit.

Seznam doporučené odborné literatury:

[1] Portnoy, M.: Virtualization Essentials. Wiley & Sons, Incorporated, 2012, ISBN: 9781118176719, 304p.
[2] von Hagen, W.: Professional Xen Virtualization, Wrox Press, 2008, ISBN: 978-0470138113, 405 p.
[3] Stagner, H.: Pro Hyper-V. Apress, 2009, ISBN: 9781430219088, 425 p.
[4] Haletky, E.: VMware ESX and ESXi in the Enterprise: Planning Deployment of Virtualization Servers. Pearson Education, 2011, ISBN: 9780137058860, 576 p.
[5] Lowe, E.: Mastering Vmware Vsphere 5. Sybex, 2011, ISBN: 9781118180129, 768 p.
[6] VMMark VIrtualizations Benchmarks. [online] [cit. 2013-11-27]. Dostupné z WWW: http://www.vmware.com/products/vmmark/

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Moravec, Ph.D.**

Datum zadání: 01.09.2013

Datum odevzdání: 07.05.2014

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: 29. července *2014*

.....
podpis studenta

Poděkování

Rád bych poděkoval Ing. Pavlovi Moravcovi za užitečné rady a motivaci během psaní této práce. Poděkování také patří všem kolegům ze společnosti DAT s.r.o. Za podporu během mých studií.

Abstrakt

Virtualizace je jednou z oblastí, která v současnosti zažívá výrazný rozmach v datových centrech i u koncových uživatelů. V bakalářské práci se zabývám problematikou pro automatické nasazení testů ve virtuálních strojích. Popíšu druhy virtualizačních technik spolu s jejich klady a zápory. Provádím jejich porovnání a v neposlední řadě otestuji vybrané virtualizační řešení. Vytvořím testovací platformu pro benchmarking různých komponent PC jako jsou: pevný disk, operační paměti, grafické výpočty a síťová propustnost. Vyhodnotím výsledky testů a doporučím konkrétní řešení pro typické scénáře nasazení.

Klíčová slova

Virtualizace, virtuální počítač, Hypervisor, benchmarking, VMware EsXi, Linux, KVM, Citrix XenServer, Terminál, PC

.

Abstract

Virtualization is one of the areas that are currently experiencing a significant boom in data centers even end users. In the thesis deals with the issue for automatic deployment testing in virtual machines. Describe the types of virtualization techniques along with their pros and cons. Doing comparing them and finally I test the selected virtualization solutions. Create a test platform for benchmarking various PC components such as hard disk, memory, graphics computing and network throughput. Will evaluate the test results and recommend specific solutions for typical deployment scenarios.

Key words

Virtualization, virtual machine, Hypervisor, benchmarking, VMware ExXi, Linux, KVM, Citrix XexServer, Terminal, PC

Seznam použitých zkratek a symbolů

Symbol / Zkratka	Jednotky	Význam symbolu
CPU	Ghz	Procesor
DomU		Neprivilegovaná doména
Dom0		Privilegovaná doména
Mb/s	Megabity za sekundu	Síťová propustnost
NAT		Překlad síťových adres
OS		Operační systém
PC		Počítač
RAID		Vícenásobné diskové pole
SCP		Secure copy
SSH		Secure Shell
UTP		Síťový kabel
VM		Virtuální stroj

Obsah

1 Úvod

Tématem této bakalářské práce je virtualizace operačních systémů. V poslední době se slovo virtualizace začalo kolem nás objevovat čím dál tím více. Na přechod do virtualizovaného prostředí láká především úspora nákladů za energie, místo v serverovnách, vyšší využití hardwarových prostředků, zjednodušená správa a mnoho dalších důvodů. Také bychom si ale měli uvědomit, že virtualizace s sebou nese i jistá rizika, proto je na místě si pečlivě zanalyzovat situaci, popřípadě diskutovat nasazení virtualizace s odborníkem. Nejprve budou vysvětleny základní pojmy, principy a fungování virtualizace. Dále budou představeni tři největší výrobci virtualizačních technologií a to Citrix, VMware a KVM, jejich produkty vSphere EsXi, a XenServer.

Jejich srovnání bude jak teoretické, tak i praktické za pomocí provádění výkonnostních testů, které se budou orientovat na výkon procesorů, operační paměti, grafických výpočtů a síťové propustnosti. Jako virtualizovaný (hostovaný) operační systém bude použita linuxová distribuce Ubuntu, která byla na začátku tvorby bakalářské práce k dispozici verzi 13.10, a to v prostředí KVM, Xen a Vmware. Ubuntu bude použito i jako nevirtualizovaný hostující systém v Xen a KVM. Je to z důvodu srovnatelnosti při výkonnostních testech, aby virtualizované systémy byly co nejvíce stejné. V tom případě budou rozdíly ve výkonu způsobeny pouze výkonem virtualizačního softwaru.

V dnešní době nám výrobci procesorů podávají pomocnou ruku v podobě podpory virtualizace už v samotném procesoru, nalezneme ji v drtivé většině nabízených produktů. V případě firmy Intel je to označení Intel-VT, u AMD je označení AMD-V. Více bude rozebráno v následující kapitole. Přínosem bakalářské práce bude vysvětlení, jak samotná virtualizace funguje, jaké jsou její možnosti a zjištění, jak si různé verze virtualizačních technologií povedou v reálných testech. Díky tomu budeme moci určit, která virtualizační technika si vede lépe a ve kterém směru než ostatní.

Nejdříve před samotným srovnáváním těchto řešení budu popisovat, co taková virtualizace znamená, co je pod tímto pojmem myšleno a v jakých spojitostech se používá. Popíši několik nejtypičtějších druhů použití virtualizace a budu se na nich snažit demonstrovat výhody virtualizace oproti standardnímu řešení, kdy na jednom hardware běží jeden operační systém. Postupně nasadím virtuální stroje v Xenu, VMware a nakonec v KVM. Ve všech řešeních provedu instalaci virtuálního (hostovaného) systému jako plně virtualizovaného. Je to z důvodu, aby si mohl každý uživatel nainstalovat svůj neupravený operační systém a je to taky jediné stejné společné řešení všech tří virtualizačních platforem.

2 Virtualizace

Základní vlastností virtualizace je oddělení software od fyzického hardware, to znamená, že vzniká abstraktní vrstva, která nám umožňuje, že nejsme závislí na konkrétním hardwaru a také umožňuje i několik takovýchto software spouštět zároveň. Pro příklad uvedu námi dobře známou situaci, kdy máme na jednom fyzickém počítači dva operační systémy: Linux Ubuntu a Windows XP. Pracujeme-li na Windows XP a chceme z nějakého důvodu ihned pracovat na Ubuntu, tak nám nezbývá nic jiného, než počítač restartovat a „nabootovat“ do našeho Ubuntu.

S pomocí virtualizací je tomu jinak. Oba operační systémy můžeme mít spuštěné současně, když budeme chtít pracovat na tom druhém, tak jednoduše se do něj „přepneme“, tím ušetříme čas strávený ukládáním rozdělaných prací, restartování počítače a následným „bootováním“. Serverová virtualizace začala tak, že virtualizační nástroje pracovaly nad známými operačními systémy, jako jsou např. Windows, Linux. Operační systém ale má vždy vyšší režii, než jednoúčelový „tenký“ operační systém vyvinutý a vyladěný pro samotnou virtualizaci. Obsah této kapitoly čerpá z [14].

Tenký operační systém se označuje jako hypervisor. Hypervisor je zodpovědný za rozdělování výpočetního výkonu, management paměti a management I/O operací. Je to mezivrstva mezi fyzickým hardware a operačním systémem. Existuje mnoho metod jak hypervisor implementovat, např. společnost VMware sází na hypervisor nainstalovaný přímo na hardware, nebo Xen [1], který má hypervisor již zabudovaný v jádře operačního systému Linux. O tom bude popsáno v následující kapitole. V typech virtualizací se používají tři typy vitualizačních metod: emulace, plná virtualizace a paravirtualizace..

2.1 Emulace

Základním principem emulátoru je překlad strojových instrukcí hostovaného systému na strojové instrukce hostitelského stroje. Emulátor je software, který umožní běh programů Díky tomu není nutné hostovaný systém nijak upravovat a je možné takto provozovat i aplikace pro jinou architekturu než má samotný hostující systém. Emuluje se procesor včetně registrů, dále se emuluje paměť ROM cílové platformy a zbytek hardware. I přes různé optimalizace (jednou přeložené úseky aplikace se ukládají do paměti, takže je není třeba při příštím volání znovu překládat) se jedná o nejméně efektivní způsob virtualizace. Na druhou stranu je to jediný způsob jak virtualizovat jinou architekturu. Lze i emulovat mnohaprocesorový stroj na počítači s jedním procesorem a podobně. K nejznámějším emulátorům patři BOCHS,Wine, PearPC. Obsah této kapitoly čerpá z [15].

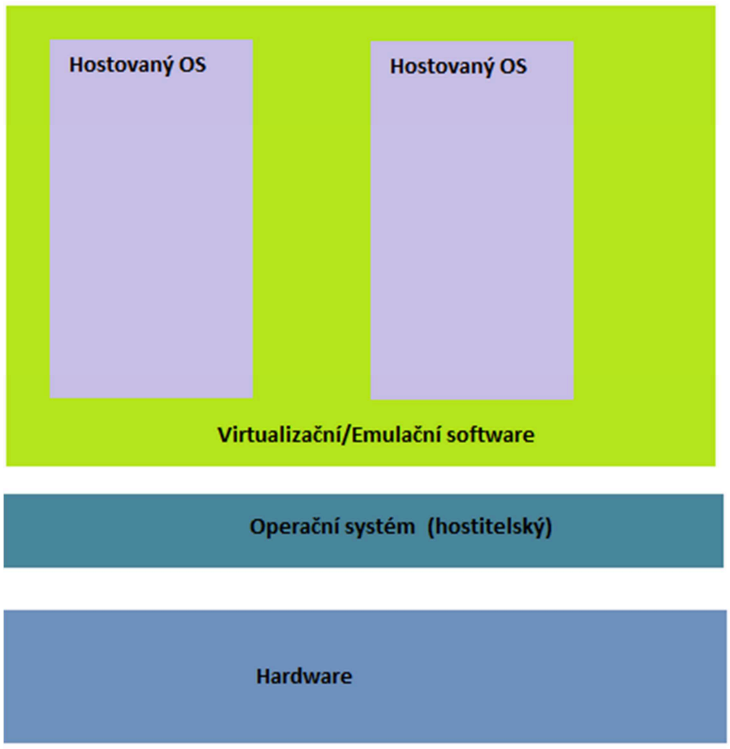
Výhody:

- možnost na libovolné platformě spustit systém a aplikace libovolné jiné (danou aplikací podporované) platformy.
- Virtualizační software je obyčejnou aplikací, běží i bez administrátorských práv.

- Hostitelský ani hostovaný systém nemusí být nijak upraveny.

Nevýhody:

- Nevýhodou je přidaná režie, protože veškeré požadavky na simulovaný systém musejí být přeloženy pro systém hostitele a po provedení operace výsledky přeloženy nazpět.

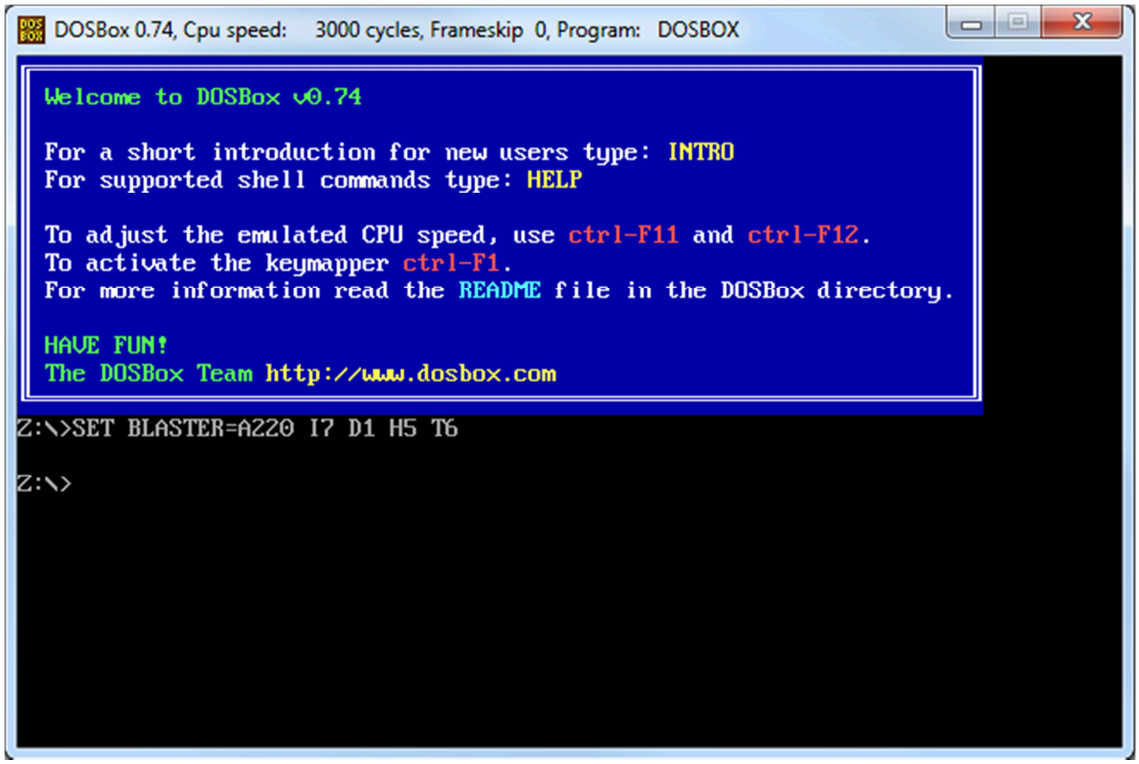


Obrázek 1.1: *Schéma emulace*

Příkladem může být aplikace DOSBox [5] . DOSBox je emulátor emulující úplné prostředí systému MS-DOS. Hardware a software jsou od sebe odděleny, je možné dokonce zavádět i jiné systémy v DOSBoxu (obsahuje vlastní zavaděč). Prioritou autorů aplikace DOSBox je však poskytnout prostředí co nejvíce kompatibilní s MS-DOS určené k provozu starých her vytvořených pro počítače s procesorem Intel 286–486 včetně matematického koprocesoru. Můžeme se setkat i s programy, které se používají pro různá účetnictví. Takovým programem je např. Majetek.exe, který už na žádném novějším systému nespustíme. Jednou z možností je jej spustit v emulátoru DOSBox. V hostitelském operačním systému Windows si můžeme namapovat USB tiskárnu hp2050 na port LPT1 pomocí příkazu:

```
net use LPT1: \\pocitac\hp2050
```

Kde „pocitac“ je název našeho počítače a „hp2050“ je název tiskárny kterou chceme mapovat, nalezneme jej ve správci zařízení. Port LPT1 už dobře zná DOSBox, tímto máme cestu otevřenou k tisknutí tiskových úloh z programu Majetek.exe.



Obrázek 1.2: *Aplikace DOSBox*

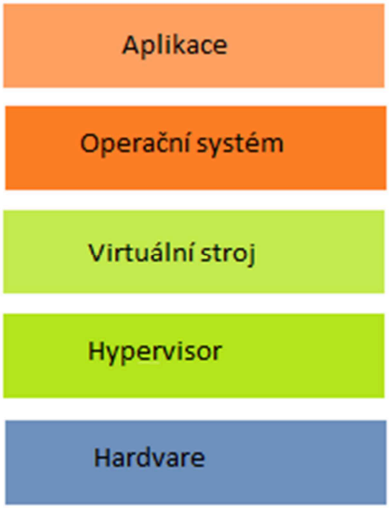
2.2 Plná virtualizace

Virtualizujeme-li důsledně všechny součásti počítače, hovoříme o tzv. *plné virtualizaci* (full virtualization)[3]. Hostovaný operační systém, ani aplikační programy nepotřebují žádné modifikace. Jedná se v podstatě o ideální stav, kdy dochází k plnému oddělení fyzické vrstvy, veškeré programy běží pouze na virtuálním hardware a přístup k fyzickému vybavení je vždy zprostředkován. Na serverech se ve většině případů setkáme s plnou virtualizací. Například pokud poskytovatel nabízí hostingové služby, bude je určitě nabízet pro širší spektrum operačních systémů, to umožňuje pouze plná virtualizace. Hostovaný stroj je emulován pomocí virtualizačního hardware. Procesor se neemuluje (z toho plyne, že platformy musí být shodné) a hostované operační systémy i aplikace běží v nativním režimu a tedy s plným výkonem. Kapitola čerpá z [20].

Problém nastane jen v situaci, kdy se hostovaný systém pokusí přistoupit k hardware. Takový požadavek je odchycen, upraven (čtení z virtuálního disku je třeba převést na čtení určitého místa na disku fyzickém a podobně) a následně předán ke zpracování hypervisorem. Výsledek bude odeslán stejným způsobem zpět do hostovaného systému.

Tímto se plná virtualizace vyznačuje. Veškeré vstupně-výstupní operace musí být zpracovány virtualizační vrstvou, než budou provedeny. Novější procesory Intelu a AMD mají speciální technologie (Vanderpool resp. Pacifica), které v tomto směru usnadňují programování virtualizačního software, ale nepřenášejí téměř žádný nárust výkonu. Výhodou tohoto druhu

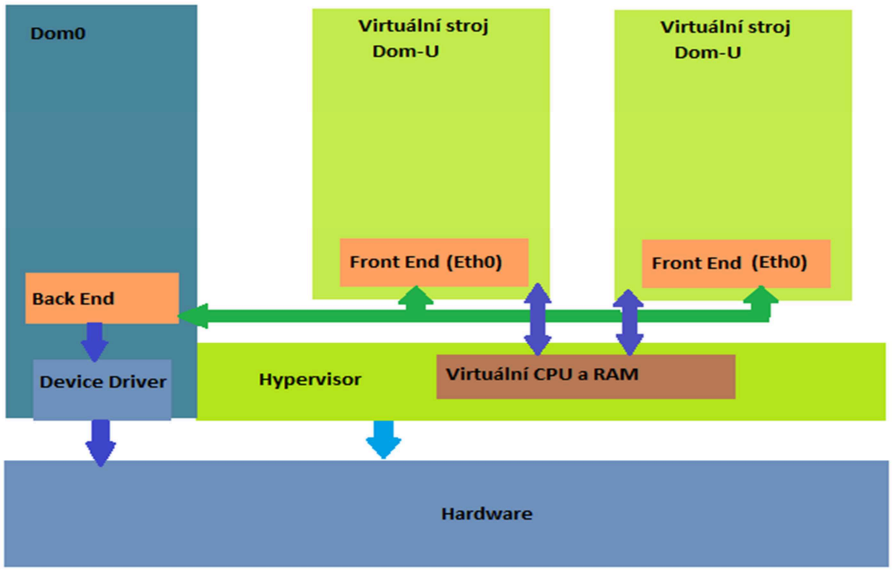
virtualizace je především možnost spustit libovolnou aplikaci schopnou běhu na fyzickém hardware bez nutnosti úprav aplikace. Nejznámější aplikace využívající plné virtualizace: VMware, Xen, KVM.



Obrázek 1.3: *Schéma plné virtualizace*

2.3 Paravirtualizace

Zatímco u plné virtualizace simulujeme hostovaným virtuálním systémům veškeré prostředky hostujícího serveru, u paravirtualizace je tomu jinak. Paravirtualizace se nesnaží simulovat veškeré prostředky, místo toho nabízí speciální rozhraní API, pomocí kterého komunikují hostované systémy s hardware skrze hostitelský systém. Jedině hostující systém (dom0) má přístup k hardware.

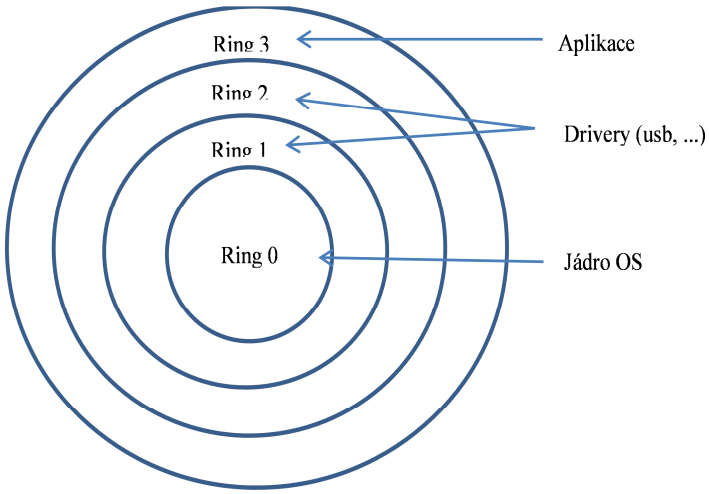


Obrázek 1.4: *Paravirtualizace Xen*

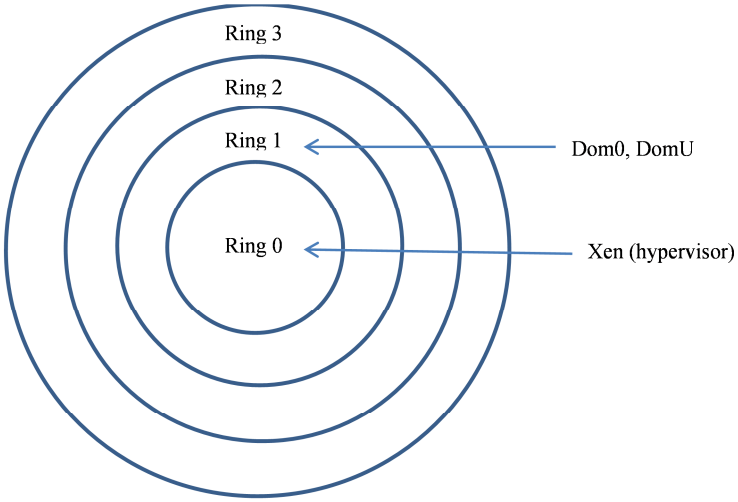
Zjednodušeně si můžeme paravirtualizaci představit jako jednoduchý kód, který předá požadavek na manipulaci s hardware do hostitelského systému (dom0) a sám s hardware nepracuje. Touto cestou jde virtualizační technika Xen. Pro tuto metodu je nutná úprava hostovaných a hostitelských systémů, tudíž nemůžeme použít proprietární software, jako jsou např. Windows, ale jsme odkázáni na operační systémy s otevřeným zdrojovým kódem, jako je Linux. Další podmínkou paravirtualizace je rozšíření úrovní ochrany, tzv. "rings". Je to proto, že nám přibyla další komponenta a tou je samotný hypervisor (virtuální monitor). Virtuální monitor musí běžet na nejvyšším stupni ochrany. Na stejné úrovni však nemůže automaticky běžet hostující operační systém, protože by mohl ovlivnit stav virtuálního monitoru. Další problém spočívá v ochraně operačního systému před běžícími uživatelskými programy.

Pokud bychom měli jen dvě úrovně ochrany (privilegované a neprivilegované), musel by operační systém virtuálního počítače pracovat neprivilegovaně, tím by však byl vystaven ohrožení ze strany aplikací. Najednou máme tři komponenty, které nesmějí pracovat ve stejných úrovních. Je to z důvodů bezpečnosti. Paravirtualizace je tak možná jen díky tomu, že konkrétní procesory podporují více úrovní ochrany.

Procesory Intel mají definovány 4 úrovně ochrany. Na nejvyšším stupni ochrany (ring 0) běží tradičně operační systém, uživatelské programy běží s nejnižším stupněm ochrany (ring 3). Různé drivery USB sběrnic, síťových karet a dalšího hardware jsou umístěny v kruhu 2, viz obrázek 1.5. Pokud použijeme paravirtualizaci, pak virtuální monitor, neboli hypervisor, pracuje na nejvyšším stupni ochrany, tj. ring 0. Operační systém virtuálního počítače se posune o jeden stupeň níže, do okruhu 1 a aplikační programy běží stále s nejmenší ochranou, jak je znázorněno na obrázku 1.6. Operační systém tímto má stále vyšší úroveň ochrany než aplikace. Ale už nemůže provádět operace, které vyžadují privilegovaný přístup. Úrovně ochrany však můžeme využít i místo výše zvýšené modifikace privilegovaných instrukcí - necháme operační systém ve virtuálním počítači provádět všechny instrukce, pokud však bude chtít provést "zakázanou" operaci tj. takovou, na kterou teď nemá dostatečná oprávnění, pak dojde k přerušení a řízení převezme virtuální monitor. Kapitola čerpá z [20]



Obrázek 1.5: *Úrovně ochrany bez virtualizace*

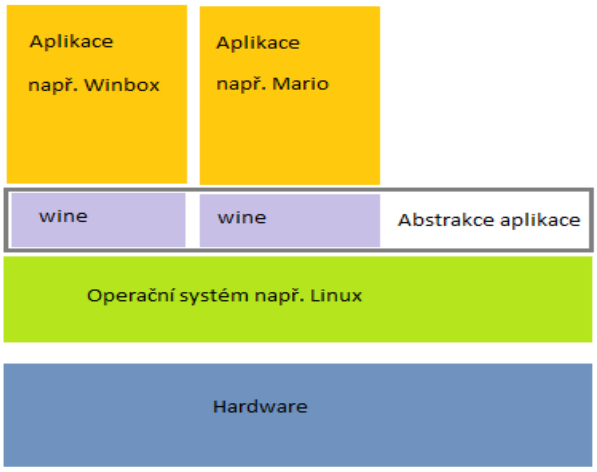


Obrázek 1.6: *Úrovně ochrany s virtualizací Xen*

2.4 Aplikační virtualizace

Aplikační virtualizace se liší od předchozích virtualizací tím, že není potřeba komplikovaně instalovat hypervisor na železo, nebo jako součást operačního systému. Zde se jedná o aplikaci, kterou si můžeme stáhnout z internetu a nainstalovat jako běžnou aplikaci.

Aplikační virtualizace se používá především pro programy, které jsou choulostivé na jakékoliv změny v systému, vzájemnou kompatibilitou s ostatními aplikacemi a kompatibilitou s operačními systémy. Tohle je a bude neustálým problémem v mnoha případech. Pomocí aplikační virtualizace vytvoříme malou mezivrstvu mezi operačním systémem a aplikací, která obsahuje pouze potřebné data ke spuštění aplikace (registry, soubory). Příkladem může být aplikace Wine [6] zobrazená na obrázku 1.7.



Obrázek 1.7: *Aplikační virtualizace*

3 Příprava

Pro srovnání výkonů virtualizačních produktů provedu sérii benchmarků, které nabízí vytvořená benchmarkovací platforma. Tímto zjistím rozdíly, jak si která virtualizace vede a ve kterém šmeru. Unixbench nabízí benchmarky výpočtů práce čísel bez a s plovoucí desetinnou čárkou, práci se zápisem a čtením z pevného disku, práce s operační pamětí a s vláknovou komunikací. Utilita Iperf slouží k měření síťové propustnosti, pomůže mi poodhalit jak hypervisor zprostředkovává přístup k fyzickému hardwaru. Utilita Glmark2 pomůže zjistit hodnoty grafických výpočtů ve virtualizovaném systému, stejně tak i v nevirtualizovaném. Nejdříve provedu benchmarking v nevirtualizovaném stroji, získám tak hodnoty maximálního využití hardware. Hodnoty následně budu porovnávat s hodnotami benchmarků virtualizovaných strojů. Bude to jistě přehlednější a můžu následně říci která virtualizace, resp. hypervizor je lepší a v jaké části benchmarků, nebo která virtualizace se nejvíce výkonnostně blíží k nevirtualizovanému systému.

Popíši instalaci jednotlivých virtualizačních produktů včetně jejich nastavení.

3.1.1 Citrix XenServer

První virtualizační nástroj jsem vybral od citrixu XenServer ve verzi 6.2.0 který je možné stáhnout ze stránek výrobce. Instalace je rychlá a intuitivní, v průběhu vyžaduje zadání několika informací jako je heslo pro účet správce, země ve které probíhá instalace a nastavení síťové komunikace. Na stroji kde je nainstalován XenServer nenajdeme kvalitní, plnohodnotné GUI kterým můžeme nastavovat a spravovat virtuální stroje. Vše se nastavuje síťově přes aplikaci, kterou si stáhneme s pomocí webového prohlížeče, v něm musíme do adresního řádku zadat IP adresu xen serveru, kterou jsme nastavili při instalaci. Případně můžeme IP adresu nastavit na serveru, který má GUI podobné biosu [21]. Stáhneme si aplikaci XenCenter, následně nainstalujeme a spustíme. Po spuštění aplikace se zobrazí okno do kterého musíme zadat IP adresu XenServeru, jméno a heslo administrátora. Pomocí xencenteru, který je velice dobře navržen, aby se v něm vyznal i naprostý laik, nastavím virtuální stroj a nainstaluju do něj Ubuntu. Pro síťovou komunikaci nastavím režim bridge aby byl virtuální stroj na síti vidět. Benchmarkování je popsáno v další kapitole.

3.1.2 VMware EsXi 5.5

Produkt značky VMware EsXi 5.5 je pouze v trial verzi na 60 dní. To ale nevadí, protože po uplynutí 60 dní bude stále funkční, pouze budou některé funkce omezeny, nebo vypnuty. Uživatel pro získání produktu EsXi se musí registrovat na stránkách výrobce, aby si jej mohl stáhnout. Následná instalace probíhá obdobně jako u xenserveru, nastavení IP adresy serveru, administrátorské heslo a jméno. Po zadání IP adresy do webového prohlížeče se objeví stránky serveru kde je odkaz ke stažení VMware vSphere Client se kterým můžeme provádět pokročilé administrátorské práce. Nastavil jsem stejně jako v předchozím virtualizačním řešení nový virtuální stroj a do něj nainstaloval Ubuntu, stejně tak síťový režim bridge. Následná

komunikace s virtuálním OS je prováděna pouze síťově přes SSH klienta. Stejně tak ovládání benchmarkovací platformy.

3.1.3 KVM

Tento virtualizační nástroj je typu plné virtualizace. Instaluje se to hostitelského OS. Nejdříve musíme zjistit jestli náš procesor obsahuje podporu plné virtualizace. Pracuji pouze v operačním systému Linux Ubuntu, proto uvedu postup, jak podporu CPU zjistit v něm.

Spuštěním příkazu:

```
cat /proc/cpuinfo | egrep --color=always 'vmx|svm'
```

zjistíme jestli je podpora virtualizace obsažena v CPU červeným zbarvením vmx, nebo svm v odpovědi příkazu. Jestli se tak stane, můžeme přistoupit k dalšímu kroku, tím je instalace samotných balíčků pro virtualizaci.

```
apt-get install ubuntu-virt-server python-vm-builder kvm-ipxe
virt-manager bridge-utils
```

Po nainstalování balíčků musíme přidat uživatele do skupiny libvirtd zadání příkazu:

```
sudo adduser soumen libvirtd
```

Nyní už jen stačí spustit aplikaci virtmanager, v ní vytvořit nový virtuální stroj s požadovanými parametry jako jsou velikost virtuálního pevného disku, operačních pamětí, počet CPU a síťové propojení. Nainstalujeme Ubuntu pomocí CD, nebo ISO souboru a můžeme benchmarkovat.

4 Testovací soustava

4.1 Příprava pro benchmarking

Po nainstalování virtuálního stroje jsem nastavil síťové připojení do režimu bridge [11] aby bylo možné se s virtuálním systémem spojit pomocí TCP/IP protokolu bez jakýchkoliv problémů, toto nastavení jsem provedl u všech virtualizačních produktů. Tímto režimem umožňujeme vidět virtuální stroj v síti jako další PC. Není zde použit překlad síťových adres NAT. Komunikace mezi virtuálním strojem a běžným PC v síti je zprostředkována hypervisorem. IP adresa virtuálního stroje je nastavena tak, aby byla v rozsahu sítě a bylo možné se na stroj připojit. Pomocí SCP protokolu jsem zaslal mnou vytvořenou benchmarkovací platformu na virtuální systém Ubuntu následným příkazem v terminálovém okně na svém PC:

```
#sudo scp bencher.sh soumen@192.168.88.55:
```

Tento postup je neměnný pro všechny tři virtuální platformy. Následně se připojím pomocí SSH programu [12] na virtuální stroj:

#sudo ssh -l soumen 192.168.88.55

Potom už jen stačí spustit platformu jako běžný skript:

#sudo ./bencher.sh

Platforma v prvním kroku zjišťuje, s jakými právy byla spuštěna, pokud nebude spuštěna pomocí příkazu sudo, nebo uživatele s rootovskými právy, vypíše chybovou hlášku a ukončí se.

Zdrojové kódy benchmarkovací platformy jsou obsaženy v přiloženém CD. Návod jak pracovat s benchmarkovací platformou je popsán v následující kapitole.

Všechny tři virtualizační platformy Xen, VMware a KVM budu postupně nasazovat na fyzický PC které má následující parametry:

Tabulka 1.1: *Fyzické parametry stroje*

Procesor	*Intel Core 2 Duo CPU E8400 3.00GHz x2, socket 775, L2 6MB*
Grafická karta	*ATI Radeon HD 7770 GHz*
Operační paměť	*4GB (2x 2GB DDR2 1066MHz)*
Síťová karta	*D-Link Systém Inc DGE-528T Gigabit eth. adapter*
Základní deska	*Gigabyte GA-EP35-DS3L*

Pro virtuální PC jsem u všech virtualizačních produktů nastavil stejné hardwarové parametry obsažené v následující tabulce, je to z důvodů lepší srovnatelnosti produktů, výsledné hodnoty benchmarků se budou vztahovat pouze k hypervisorům.

Tabulka 1.2: *Parametry virtuálních strojů*

Počet CPU	2
Operační paměť	4GB
Hard disk	20GB
Síťová karta	Sdílená, režim bridge

4.2 Testovací software

Pro testovací software jsem se rozhodl použít programy, jako jsou: Unixbech, Iperf a glmark2. Jsou to programy lehce dosažitelné v linuxových distribucích. Níže popíšu stručně jejich vlastnosti.

4.2.1 UnixBench

UnixBench je původní BYTE UNIX benchmarkovací soustava.

Účelem Unixbench [4] je poskytnout základní ukazatel výkonu unixových systémů. Obsahuje více testů, které se používají k testování různých aspektů výkonnosti systému. Tyto výsledky testů jsou pak ve srovnání se skóre s výchozím systémem produkovány jako hodnoty indexu, které jsou obecně snazší pro pochopení než surové skóre. Celá sada hodnot indexu je pak se sumarizována pro celkový index systému. Nás nebude zajímat se sumarizován výsledek, ale jednotlivé hodnoty indexů.

Systémy s více CPU jsou také podporovány. Pokud náš systém má více procesorů, výchozím chováním je spustit vybrané testy dvakrát - jednou s jednou kopii každého testovacího programu běžícího v čase, a jednou s N kopií, kde N je počet procesorů. To je navrženo tak, aby nám umožnilo zhodnotit:

- Výkon systému při spuštění jedné úlohy
- Výkon systému při spuštění více úloh
- Zisk systému při paralelním zpracování

4.2.1.1 ***Shrnutí testů:***

- *Dhrystone:*
  Vyvinutý Reinholdem Weickerem v roce 1984. Tento test se používá k měření a porovnání výkonu počítačů. Test se zaměřuje na manipulaci s řetězci, nejsou zde žádné operace s plovoucí desetinou čárkou. To je silně ovlivněno harwarovým a softwarovým vybavením a kompilátorem. Možnosti, optimalizace kódu, vyrovnávací paměti, wait stavů, a celočíselné datové typy.

- *Whetstone:*

  Tento test měří rychlost a efektivitu operací s plovoucí desetinnou čárkou. Obsahuje několik modulů, které jsou určeny k zastupování různých operací, které jsou obvykle prováděny ve vědeckých aplikacích. Široká škála funkcí včetně sinus, cosinus, odmocnina, mocnina a logaritmus a jsou prováděny jak s celými čísly, tak s plovoucí desetinnou čárkou. Dále matematické operace, přistup k polím, podmíněným větvím a volání procedur.

- *Execl Throughput:*

  Tento test porovnává množství execl volání které mohou být provedeny za vteřinu. Execl je část rodiny exec funkcí, které nahradí stávající procesní obraz novým obrazem procesu. To a mnoho dalších podobných příkazů jsou pro funkci execve ()

- *File Copy:*

  Měří rychlost jakou lze data přenést z jednoho souboru do druhého, s použitím různých velikostí vyrovnávací paměti. Čtení, zapisování a kopírování souboru testuje zachycování počtu znaků, které lze zapsat, číst a kopírovat v určeném časovém úseku, výchozí hodnota je 10 vteřin.

- *Pipe-based Context Switching:*

  Tento test měří, kolikrát dva procesy mohou vyměňovat vzrůstající číslo integer vláknem pipe. Přepínání na bázi vláken je hodně podobné skutečným aplikacím. Testovací program založí podřízený proces, s nímž přenáší obousměrnou vláknovou komunikaci.

- *Pipe Throughput:*

  Vlákno je nejjednodušší forma komunikace mezi procesy. Vláknová propustnost je číslo počtu opakování (za sekundu) kdy proces může napsat 512 bajtů do vlákna a přečíst je zpět. Test vláknové propustnosti nemá žádný skutečný protějšek v reálném světě programování.

- *Process Creation:*

  Proces vytváření odkazuje na skutečné vytváření kontrolní procesů, bloků a přidělení paměti pro nové procesy, takže se test týká přímo šířky pásma paměti. Typicky toto

kritérium by mohlo být použito k porovnání různých implementací operačních systémů.

- *Shell Scripts:*

  Testy shellových skriptů měří, kolikrát za minutu proces může spustit a ukončit sadu jednoho, dvou, čtyř a osmi souběžných kopií skriptů, kde skript požívá řadu transformace do datového souboru.

- *System Call Overhead:*

  Tento test odhaduje cenu za vstup a opuštění jádra operačního systému, tj. režii pro provedení systémového volání. Skládá se z jednoduchého programu, který opakovaně volá na getpid (která vrací ID procesu volajícího procesu) systémového volání. Čas k provedení na tohoto volání se používá k odhadu nákladů na vstupu a výstupu z jádra.

4.2.2 Iperf

Utilita iperf [2] je známá především svou jednoduchostí měření síťové propustnosti. Podmínkou pro měření síťové propustnosti je potřeba vlastnit 2 stanice, kde první bude v režimu klient a druhá v režimu server. Po navázání spojení začne maximální možnou nebo nastavenou rychlostí posílat data. Základem ke každému měření je spustit server, k čemuž je nejjednodušší možnost použít parametr „*-s*":

```
# iperf -s
```

Na klientské stanici je potřeba zadat následnující:

```
# iperf -c <IP ADRESA SERVERU>
```

V tu chvíli se nám zahájí přesun náhodně vygenerovaných dat směrem od klienta k serveru. Po určitou dobu času (přibližně 10 vteřin) je ukončen přesun a data jsou posílána opačným směrem od serveru ke klientovi.

Na konci měření nám iperf vypíše report měření. Nebudu už dále rozebírat vlastnosti utility Iperf, bylo by to nad rámec bakalářské práce. Popíši jen nastavení utility iperf v automatizované platformě níže.

4.2.3 Glmark2

Glmark [7] je benchmark grafických výpočtů, nabízí sadu scén, které lze použít k měření mnoho aspektů OpenGL výkonu. Způsob, jakým může být každá scéna renderovaná je konfigurovatelná prostřednictvím souboru voleb. Já nechávám utilitu v default nastavení pro

srovnatelnější výsledky. Výstup z utility Glmark2 má více hodnot, nás bude zajímat celkové skóre.

4.3 Testovací platforma

Pro testování různých částí virtuálního stroje jsem použil výše uvedené utility ovládané skripty v jazyce BASH [13]. K nim patří ještě další podpůrné utility a to Gnuplot, Sshpass a Dialog.

4.3.1 Gnuplot

Gnuplot [8] je program pro generování dvou a trojdimenzionálních grafů funkcí či dat. Program běží na všech hlavních platformách a operačních systémech. Výsledek vykresluje na obrazovku, do grafického souboru popř. do textového souboru.

Program je distribuován pod svobodnou licencí, která umožňuje šíření a modifikaci zdrojového kódu. Modifikované verze mohou být šířeny jen jako patche. Program nemá nic společného s projektem GNU a jeho GPL licencí. S programem lze pracovat interaktivně, ale rovněž lze pro něj psát skripty.

4.3.2 Sshpass

Sshpass [9] je utilita sloužící k neinteraktivnímu síťovému připojení na vzdálený stroj. Je použita z důvodu automatizace v testovací platformě. Bude dále vysvětlena níže.

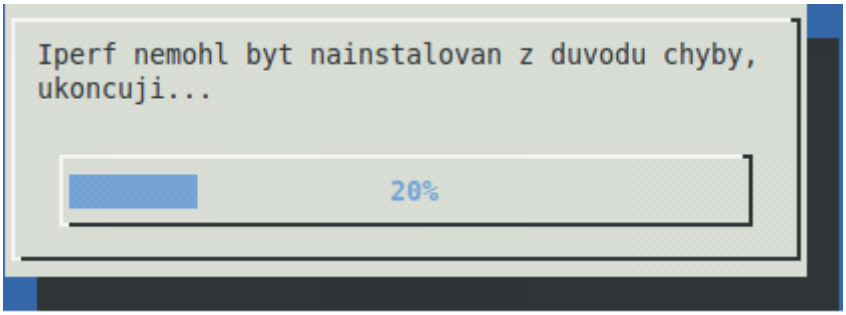
4.3.3 Dialog

Poslední utilita Dialog [10] nám nabízí interaktivní, přívětivé rozhraní, se kterým by neměl mít problémy žádný uživatel. Jedná se o terminálové, s trochou nadsázky GUI, které by se dalo přirovnat systému MS-DOS.

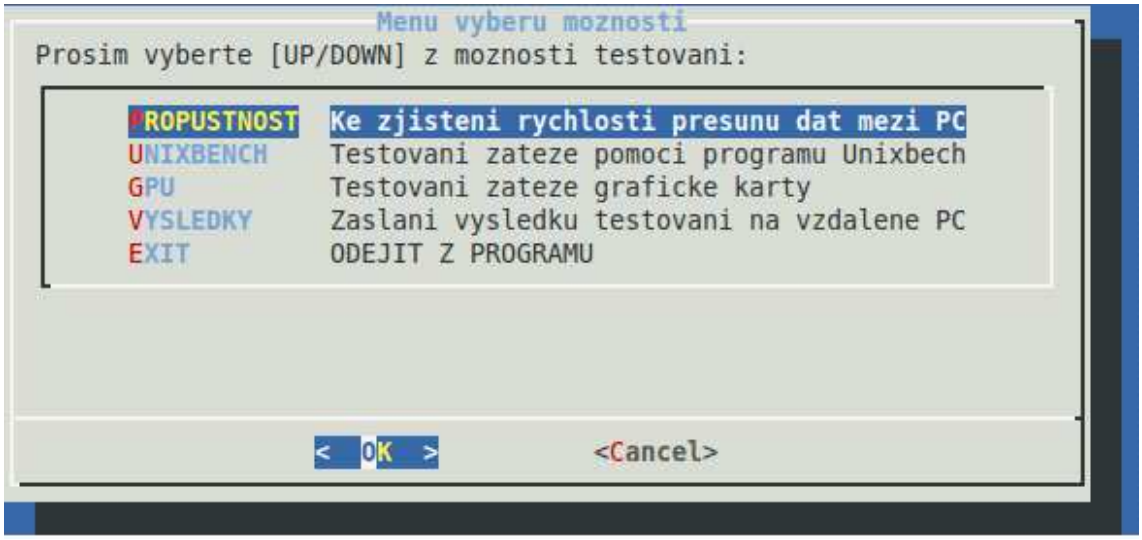
4.3.4 Benchmarkovací Platforma

Po každém spuštění bude platforma ověřovat, jestli se nacházejí utility v systému. Pokud se utilita nenachází, začne ji stahovat z internetu. Proto je na místě mít připojení k internetu při prvním spuštění platformy. Pokud ani platforma nemůže stáhnout z repozitářů utility, napíše chybovou hlášku a ukončí se, viz následující obrázek. Je tedy nutné si potřebnou utilitu stáhnou ručně, například následujícím příkazem:

```
# apt-get install iperf
```



Obrázek 1.8: *Chyba, nenalezen Iperf, ani nemohl být nainstalován.*



Obrázek 1.9: *Hlavní MENU*

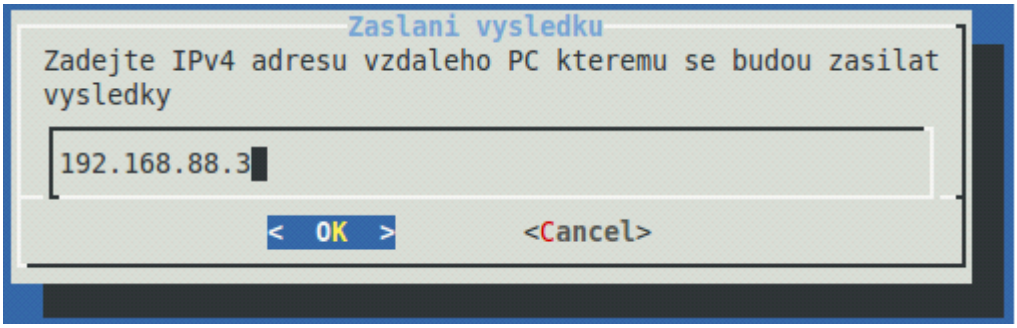
Když máme všechny potřebné utility nainstalované, dostaneme se dále do hlavního menu. Hlavní Menu (obrázek 1.9) platformy obsahuje položky, jako jsou PROPUSTNOST která používá utilitu Iperf, UNIXBENCH byla vysvětlena v předchozí kapitole, GPU která spouští Glmark2 a VYSLEDKY zasílající grafy pomocí SCP. Položka EXIT ukončuje platformu. Vybereme si položku a následně stiskem klávesy „Enter“ platforma udělá veškerou práci za nás, včetně grafů. Výjimkou jsou položky PROPUSTNOST a VYSLEDKY , první uvedená po stisku „Enter“ otevře další dialog ve kterém se platforma dotazuje na IP adresu vzdáleného PC se kterým bude měřit síťovou propustnost. Položka VYSLEDKY obsahuje taky následující dialogy, dotazuje se na IP adresu vzdáleného PC a uživatelské jméno s heslem, bude popsáno níže.

Po provedení benchmarku, nebo zaslání výsledku je uživatel vrácen do hladního MENU, kde může zvolit další benchmarky dle libosti. Výsledky v podobě grafů jsou ukládány v adresáři, kde je uložena samotná platforma/skript. V názvu souboru výsledného grafu se nachází: vysledky.<jméno benchmarku> <datum a čas vytvoření>.png. Kde <jméno benchmarku> může nabývat tří různých řetězců v závislosti na použitém benchmarku, tím jsou „propustnost“, „unixbench“ a „GPU“. Zaslání výsledků se provádí pomocí položky

VYSLEDKY, to je realizováno protokolem SCP na vzdálený stroj. Zde se používá utilita Sshpass, která umožnuje neinteraktivní zasílání dat.

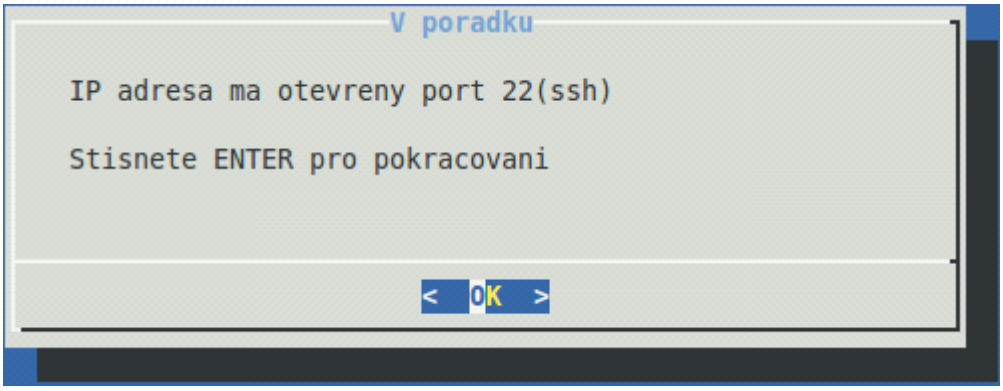
```
# sshpass -p $heslo scp vysledky* $jmeno@$ip: 2>dev/null;
```

Pro doplnění proměnných $heslo, $jmeno a $ip je uživatel postupně tázán na IP adresu vzdáleného PC, uživatelské jméno a heslo. Při správně zadaných údajích platforma zašle veškeré výsledky (grafy) na vzdálený PC, kde si jej uživatel může otevřít a prohlédnout. Následující obrázky znázorňují průběh zasílání výsledků:



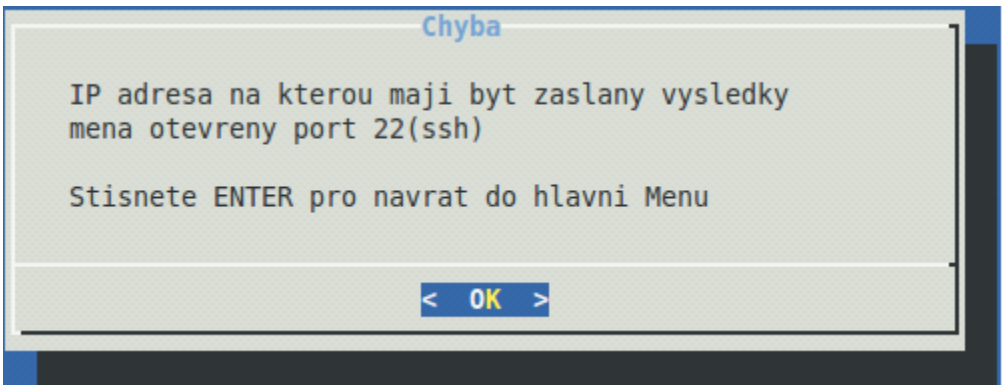
Obrázek 1.10: *Zaslání výsledků-zadání IP adresy*

Nejprve se platforma dotazuje na IPv4 adresu vzdáleného PC kterému budou zaslány výsledky. Po zadání IP adresy je adresa kontrolována jestli je validní, to znamená, že neobsahuje znaky abecedy, neobsahuje více čísel, než má mít. Pokud je návratová hodnota validace 0 tzn. TRUE, přejde k dalšímu kroku, tím je kontrola otevřenosti TCP portu 22 [19] na kterém naslouchá SSH. Pokud je opět návratová hodnota rovna nule, dialog zobrazí zprávu s kladnou odpovědí:



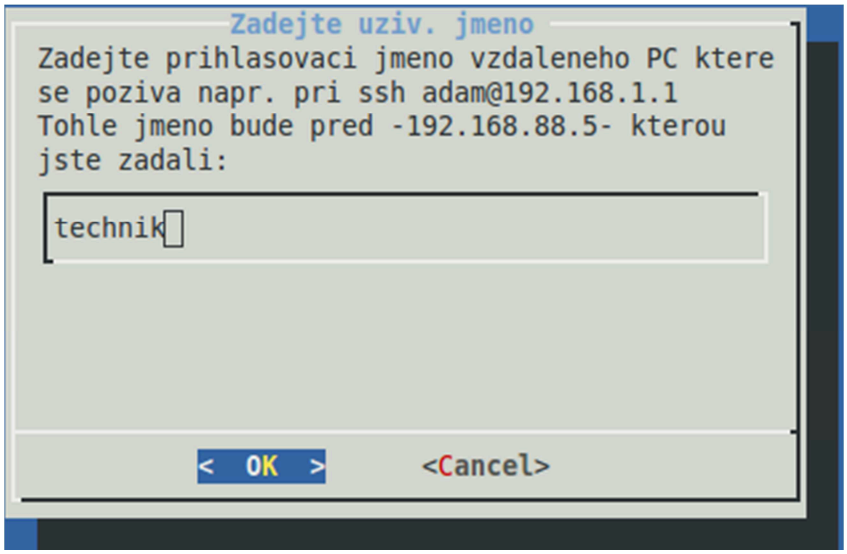
Obrázek 1.11: *Kontrola otevřenosti portu 22 - v pořádku*

Pokud bude TCP port 22 na vzdáleném PC zavřený, zobrazí se chybová hláška zobrazená v následujícím obrázku 1.12



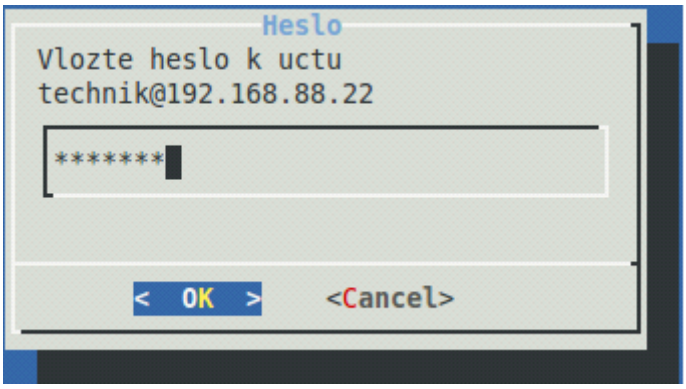
Obrázek 1.12: *Kontrola otevřenosti portu 22- chyba, není otevřen*

Po stisknutí „OK“ na chybovém hlášení je uživatel vrácen do hlavního menu, pokud proběhla kontrola portu 22 s úspěchem, je uživatel dotazován v dalším dialogovém okně, aby zadal uživatelské jméno, které se používá na vzdáleném PC. V obrázku je i uveden příklad v titulku, kde bude použito přihlašovací jméno. Viz následující obrázek:



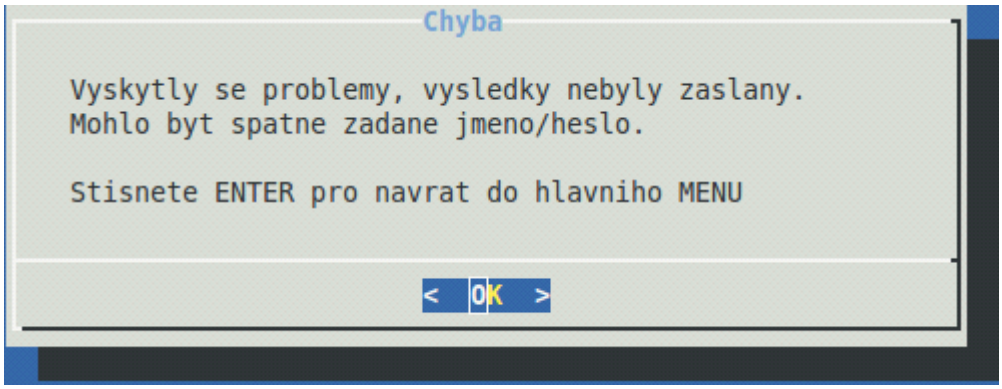
Obrázek 1.13: *Zadání uživatelského jména vzdáleného PC*

Uživatel zadá jméno, stiskne „OK“ nebo jen klávesu „Enter“, pak už je požádán jen o zadání hesla. Heslo je skryto pod hvězdičkami, jak to bývá v bezpečnosti.



Obrázek 1.14: *Zadání hesla k uživatelskému účtu na vzdáleném PC*

Jméno, heslo a IP adresa jsou zaslány do utility sshpass, která zasílá veškeré grafy na vzdálený PC. Zbývá jen kontrola odeslaných dat. To je zajištěno návratovou hodnotou utility Sshpass, pokud bude jiná než nulová, tak zasílání dat se neprovedlo v pořádku, uživatel mohl zadat špatně heslo, nebo jméno. Chybová zpráva je znázorněna na obrázku 1.15.



Obrázek 1.15: *Chyba při zasílání dat na vzdálený PC*

Když proběhne odesílání v pořádku, tak na vzdáleném PC objevíme soubory s názvem typu „vysledky.propustnost.23.01-12.07.2014.png“. V tomhle případě se jedná o benchmark síťové propustnosti, který byl proveden ve 23 hodin 1 minutu 17. 7. 2014.

5 Benchmarking komponent

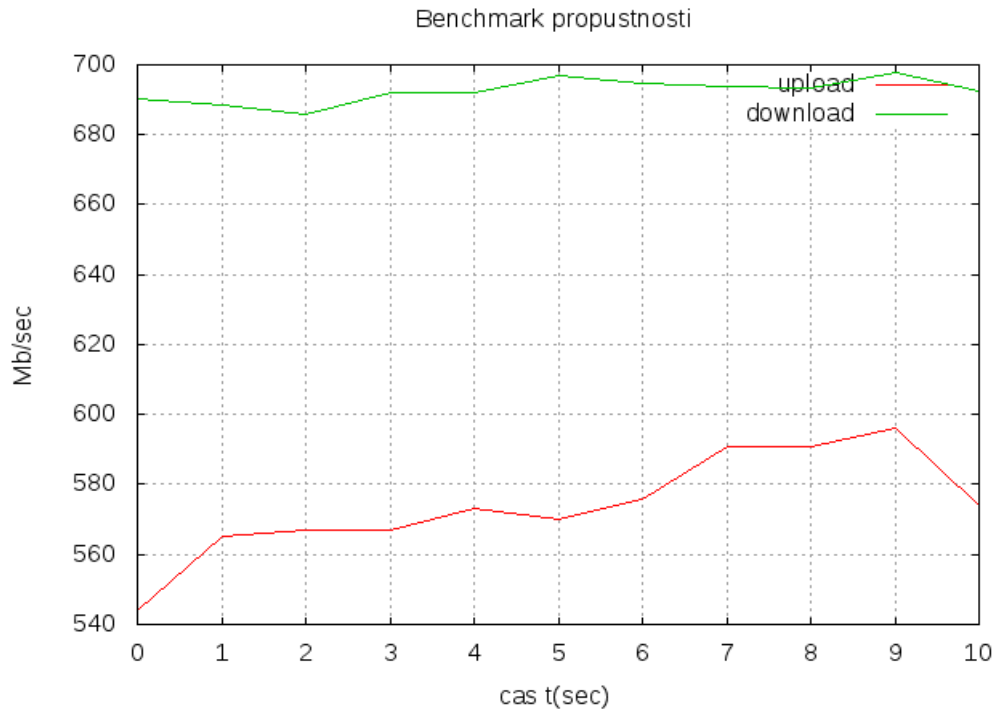
Benchmarkování komponent virtuálních strojů ve virtualizačních řešeních Citrix XenServer 6.2.0, Vmware EsXi 5.5 a KVM provedu postupně na jednom fyzickém stroji. Je to z důvodů lepší srovnatelnosti výsledků. Veškeré benchmarky budu opakovat pětkrát, aby se předešlo náhodné chybě v měření, následně je zprůměruji. Výchozí referenční hodnoty benchmarků budou odebrány ze samotného hostovaného operačního systému. Tímto dosáhnu hodnot pro následné srovnání virtualizačních řešení. Bude tak snadno zjistitelné, která virtualizační metoda si vede lépe, nebo hůře a ve kterém benchmarku.

5.1 Síťová propustnost

Pro benchmarking síťové propustnosti bude použita síťová karta značky *D-Link System Inc DGE-528T Gigabit eth. Adapter.*

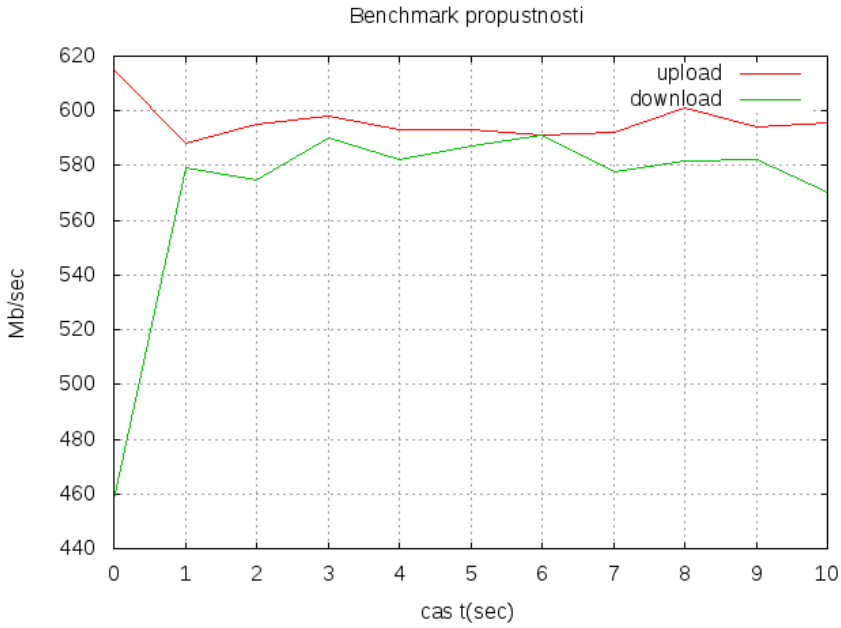
Jako druhá síťová karta vzdáleného PC bude *Intel Corporation 82567LF Gigabit Network Connection.*

Při měření síťové propustnosti nebude použit žádný další síťový prvek, jake je router, nebo switch. Mohli by ovlivňovat benchmarking. Rychlost síťových karet je nastavena na 1Gb/s. Síťové zapojení bude napřímo UTP kabelem ze síťové karty do síťové karty. Následné grafy jsou již zprůměrované a zobrazují propustnost v časovém intervalu 1 až 10 vteřin pro oba směry, download a upload.



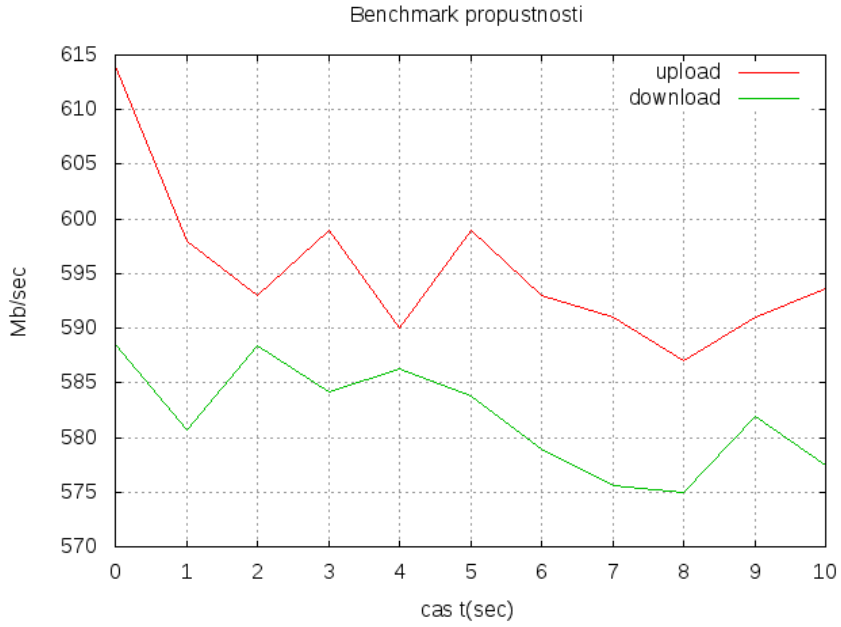
Obrázek 1.16: *Síťová propustnost hostujícího PC*

Zde v grafu 1.16 vidíme maximální možnou rychlost přenášení dat v Megabitech za sekundu. Jedná se o měření síťové propustnosti v hostujícím OS s dalším fyzickým PC.



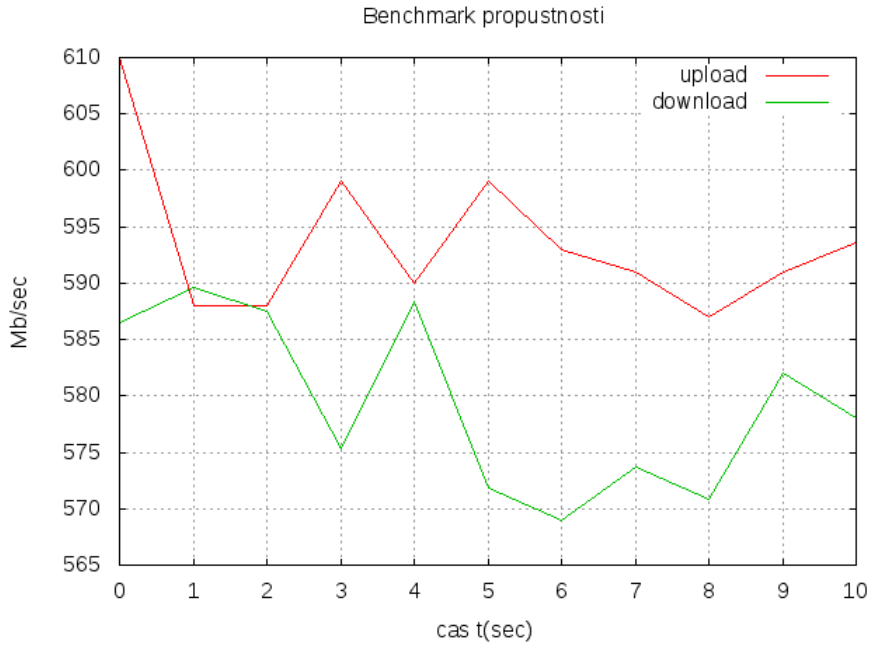
Obrázek 1.17: *Síťová propustnost hostovaného PC v XenServer*

U virtualizovaného systémů v XenServeru je vidět, že si hypervisor žádá určitou režii. Tím je ovlivněna rychlost síťové propustnosti. Rychlost ale není výrazně snížená, obyčejný uživatel by to nejspíš ani nepostřehl.



Obrázek 1.18: *Síťová propustnost hostovaného PC ve VMware*

U virtualizace pomocí VMware klesá propustnost až na polovinu maximální rychlosti 1Gb/s. Můžeme si to vysvětlit tím, že hypervisor zprostředkovává omezenější přístup k síťové kartě, než jak to bylo u XenServeru.

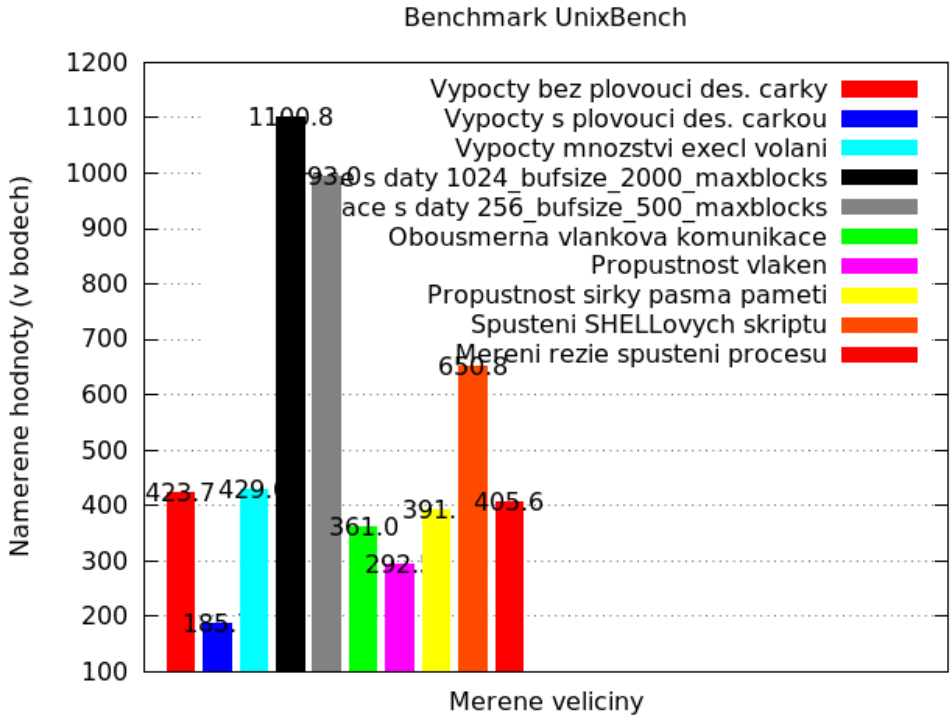


Obrázek 1.19: *Síťová propustnost hostovaného PC v KVM*

Stejně tak si vede virtualizace pomocí KVM. Síťová propustnost nevyužívá plný potenciál síťové katry, o přístup k hardwarové síťové katře se musí dělit hostovaný OS s hostitelským.

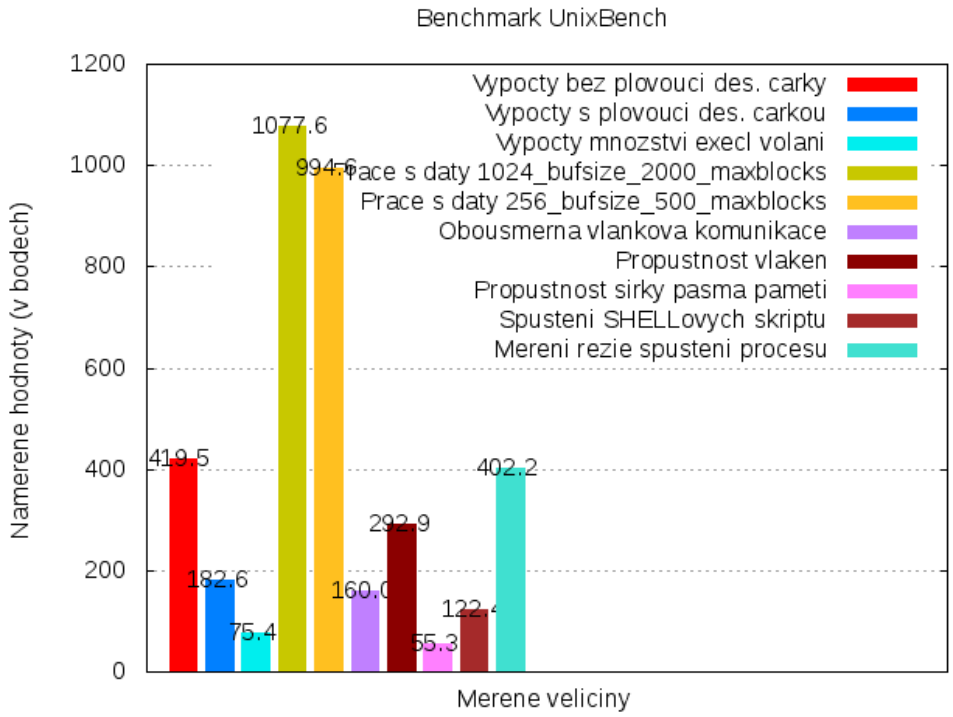
5.2 Benchmark UnixBench

Zde se zaměřím na benchmarking pomocí utility UnixBench. Ta nám benchmarkuje následující komponenty: pevný disk, operační paměti a procesor. Opět provedu benchmarking nejdříve v hostujícím OS, následně v XenServer, VMware a KVM. Grafy jsou generovány pomocí utility Gnuplot, hodnoty měřených částí jsou nešťastně umístěny na spoupcích. Nenašel jsem řešení tohoto problému.

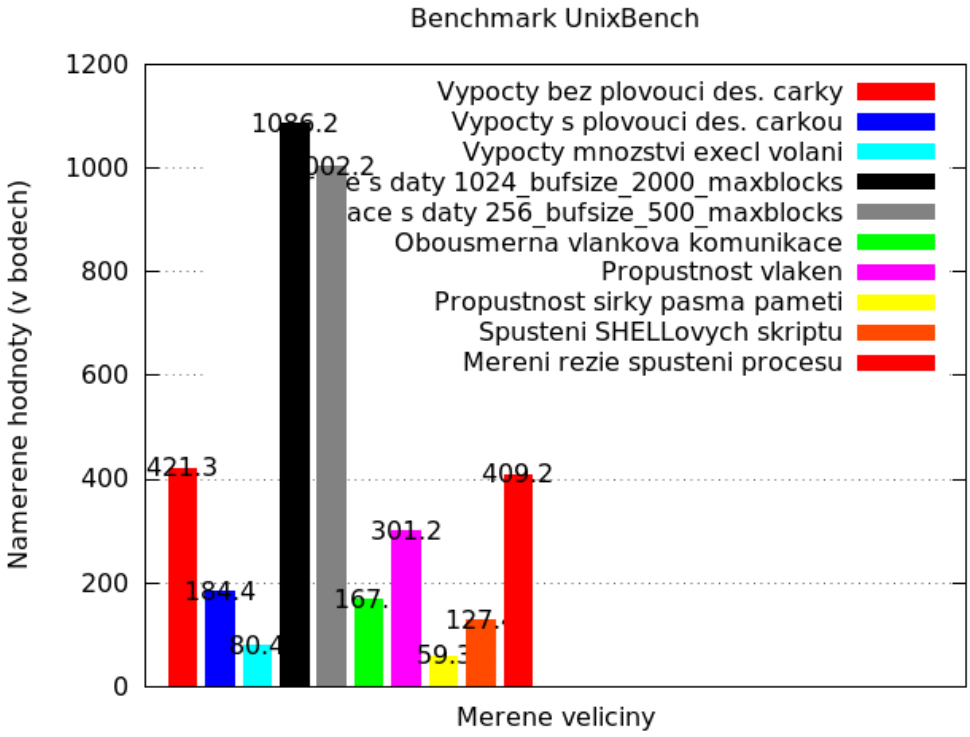


Obrázek 1.20: *UnixBench hostitelského PC*

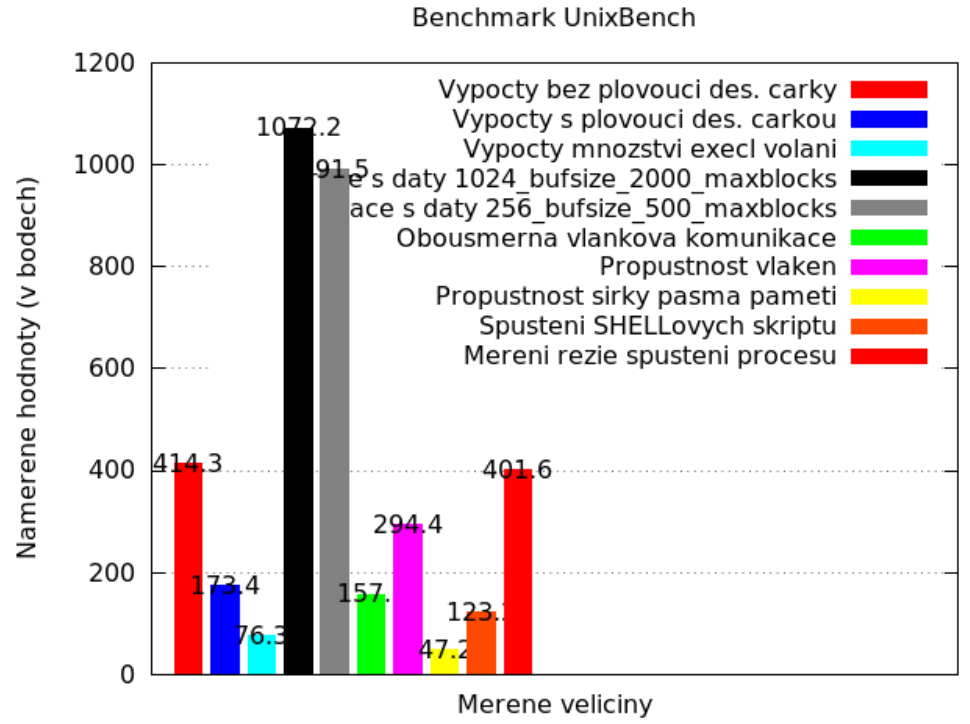
V grafu 1.20 jsou znázorněny maximální hodnoty výpočtů pro procesor, operační paměti a práce s daty na pevném disku pro hostitelský stroj.Poslouží nám to jako výchozí hodnoty pro srovnání s virtualizačními produky.



Obrázek 1.21: *UnixBench hostovaného PC v XenServer*



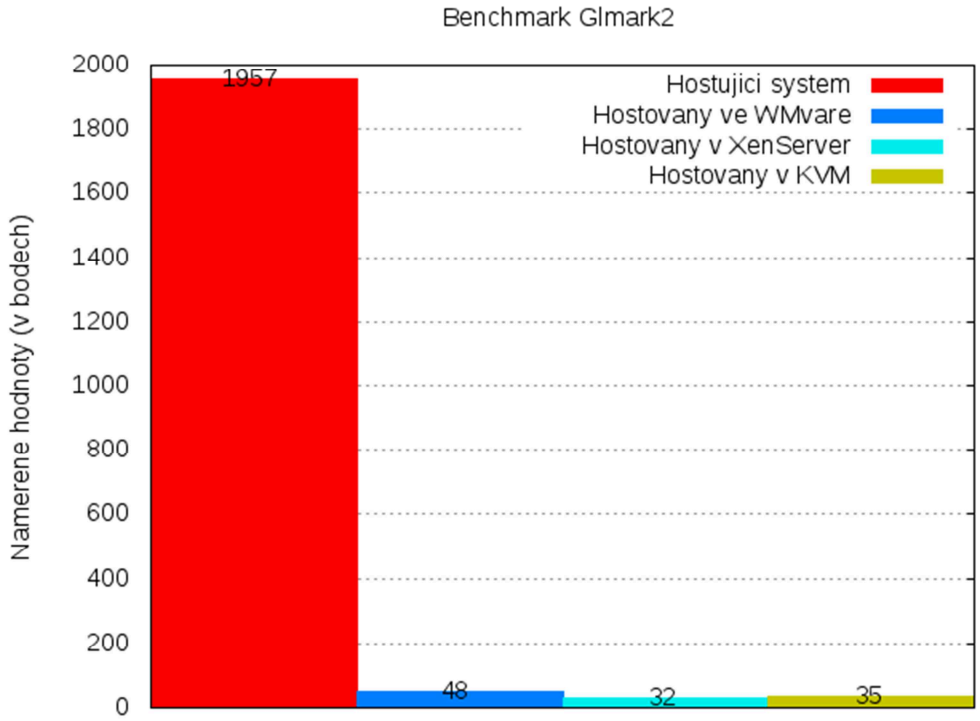
Obrázek 1.22: *UnixBench hostovaného PC v VMware*



Obrázek 1.23: *UnixBench hostovaného PC v KVM*

5.3 Benchmark Glmark2

Jak již bylo dříve popsáno, zajímá nás celkové skóre z výstupu programu Glmark2. Jedná se tedy o jednu hodnotu. Výsledkem je jednoduchý sloupcový graf, do kterého jsem sjednotil hodnoty tří virtualizačních produktů plus hodnotu benchmarku z hostujícího OS. Výsledkem je graf 1.24 znázorňující maximální výkon grafické karty hostujícího OS v porovnání s virtualními grafickými kartamy ve virtuálních strojích.



Obrázek 1.24: *Graf znázorňující maximální výkon grafické karty hostujících OS s hostovanými.*

U virtualizací jsou hodnoty zlomkem naměřené hodnoty bez virtualizace. Hypervisor měl značné vytížení a nedokázal poskytnou vyšší přístup hostovaného systému k fyzické grafické kartě. Nejlépe skončila virtualizace u WMware s 48 body, avšak v porovnání s hostitelským systémem je to zanedbatelné. Můžeme říci, že grafické výpočty u všech tří virtualizačních řešení neposkytují ani z poloviny takový výkon jako hostující systém.

6 Zhodnocení výsledků

Ve výsledných grafech je vidět, že virtualizace, respektive hypervisor má určitý vliv na naměřené hodnoty. Tudíž se hodnoty nikdy nebudou rovnat hodnotám hostujícího stroje. Pojďme se podívat blíže na naměřené hodnoty. V tabulce 1.3 jsou vypsány hodnoty benchmarků pro OS ve virtualizaci a hostujícího OS bez virtualizace. V grafickém benchmarku Glmark2 si nejlépe vede virtualizace od VMware oproti ostatním, je to ale mizivá hodnota oproti běžnému systému bez použití virtualizace. V benchmarku Unixbench si také vede lépe než ostatní virtualizace. V porovnání benchmarku UnixBench se virtualizace blíží hodnotám hostujícího OS, kromě případů execl volání, vláknové komunikace a spouštění SHELL skriptů. Podle celkového průměru hodnot je na prvním místě virtualizace od VMware s hodnotou 353, druhé místo patří virtualizace XenServeru s 346 a poslední třetí místo patří KVM 344. Hodnoty „Celkový průměr hodnot“ jsou pouze orientační průměry z naměřených hodnot. Ale i tak nám dávají dostatečný přehled, jak si která virtualizace vede. Z výsledků soudím, že ani jeden virtualizační produkt nabude vhodný pro hraní počítačových her, ani pro multimediální práce. Spíše bych se klonil ke kancelářským pracím, jako jsou práce s textovými editory, nebo pro síťové datové úložiště.

Tabulka 1.3: *Tabulka naměřených hodnot benchmarkovací platformou*

Hostující OS	XenServer	VMware	KVM		
1957	32	48	35	Glmark2	
423.7	419,5	421.3	414.3	bez plovoucí des. č.	UnixBench
185	182.6	184.4	173.4	s plovoucí des. čárkou	
429.6	75.4	80.4	76.3	execl volaní	
1100.8	1077.6	1086.2	1072.2	práce s daty 1024 buf.size	
993	994.6	1002.2	991.5	práce s daty 256 buf.size	
361	160	167.3	157.3	vláknová komunikace	
292	292.9	301.2	294.4	vláknová propustnost	
391	55.3	59.3	47.2	operační paměť	
650.8	122.4	127.4	123.1	spoštění SHELL skriptů	
405.6	402.2	409.2	401.6	měření režie spouštění procesů	
653	346	353	344	Celkový průměr hodnot	

7 Závěr

V této bakalářské práci jsem se snažil vysvětlit pojem virtualizace, její principy a fungování. Výhody virtualizace jsou především v ušetřených finančních nákladech za zakoupení nových serverů, jejich chlazení a bezesporu ušetřeným prostorem v serverovnách. Mezi nevýhody patří nevyužití veškerého potenciálu fyzického stroje. Vytvořil jsem benchmarkovací platformu která je plně automatická a přístupná pro počítače nacházející se ve stejné síti. Není potřeba mít ke zvirtualizovaným strojům připojený monitor, nebo složitě nastavovat vzdálený přístup pomocí nástrojů pro vzdálenou plochu. Benchmarkovací platforma pracuje v terminálovém okně. Umožňuje to připojení jakéhokoliv počítače v síti, který má nainstalován program SSH, tím většina linuxových distribucích. Utilita Dialog je velice užitečná pro uživatele, kteří mají problémy s orientací v Shellu. Nabízí komfortní grafické prostředí uvnitř terminálového okna pro lokálního uživatele, nebo pro vzdáleného.

Provedl jsem sadu výkonnostních testů a popsal která virtualizace je lepší než ostatní, tím je virtualizace v prostředí WMvare která byla ve všech benchmarkovacích testech o krůček lepší před ostatními virtualizacemi. Bakalářská práce by se dala rozšířit o další benchmarky úpravou zdrojového kódu benchmarkovací platformy. Například měření síťové propustnosti při maximálním vytížení CPU, měření zapisování/čtení dat o různých velikostech při zapojených discích v poli RAID.

Použitá literatura

[1] Xen, http://wiki.xen.org/wiki/Xen_Project_Software_Overview

[2] Iperf, http://openmaniak.com/iperf.php

[3] Plná virtualizace, http://en.wikipedia.org/wiki/Full_virtualization

[4] UnixBench, https://code.google.com/p/byte-unixbench/

[5] DOSBox http://en.wikipedia.org/wiki/DOSBox

[6] Wine,http://wiki.winehq.org/ImportanceOfWine

[7] Glmark2 http://afrantzis.wordpress.com/2011/12/16/glmark2-more-than-a-benchmark/

[8] Gnuplot http://www.gnuplot.info/faq/faq.html

[9] Sshpass http://www.cyberciti.biz/faq/noninteractive-shell-script-ssh-password-provider/

[10] Dialog http://www.unixcl.com/2009/12/linux-dialog-utility-short-tutorial.html

[11] Brigde http://cs.wikipedia.org/wiki/Bridge

[12] SSH http://cs.wikipedia.org/wiki/Secure_Shell

[13] BASH http://cs.wikipedia.org/wiki/Bash

[14] Hypervisor http://www.vmwarenews.cz/vmw/vmwnews.nsf/cmn/53bb1b111be5a818c12575e5005f83a6

[15] Emulace http://miho.blog.zive.cz/2008/07/typy-virtualizace/

[16] Portnoy, M.: Virtualization Essentials. Wiley & Sons, Incorporated, 2012, ISBN: 9781118176719, 304p.

[17] Von Hagen, W.: Professional Xen Virtualization, Wrox Press, 2008, ISBN: 978-0470138113, 405 p.

[18] Stagner, H.: Pro Hyper-V. Apress, 2009, ISBN: 9781430219088, 425 p.

[19] Porty http://cs.wikipedia.org/wiki/Seznam_%C4%8D%C3%ADsel_port%C5%AF_TCP_a_UDP

[20] Plná virtualizace http://www.ics.muni.cz/bulletin/articles/545.html

[21] Xenserver http://lh4.ggpht.com/jason.willey/SM7uquCv_MI/AAAAAAAAAD8/qgVP7c4PWFQ/s1600-h/xencenter5-3%5B3%5D.jpg